5-2017

Developing Sustainable Rice Production through Image Classification

Parker R. Fitzgerald
University of Arkansas, Fayetteville

Developing Sustainable Rice Production through Image Classification

An Undergraduate Honors College Thesis
In the

Department of Industrial Engineering
College of Engineering
University of Arkansas
Fayetteville, Arkansas

By

Parker Rieff Fitzgerald

Abstract

The agricultural industry's continued success is critical to our future. However, many agricultural practices—specifically rice farming—are currently based on unsustainable methods. The processes used in rice farming consume water at an unsupportable rate, and, consequently, are serious contributors to rising emission levels. Several alternative growing methods have been introduced, and one of the most promising candidates is Alternative Wetting and Drying (AWD). An obstacle encountered by proponents of these methods comes in implementation. Studies have shown AWD is more successful in fields with higher leveling precision. The research in this study works to create a stepping stone for farmers to move to a more sustainable age of agriculture. This study proposes the creation of a dynamic database containing fields and their levelling technique and employs machine learning image classification methods to begin this work. If this database were to be created, effective programs could be developed to incentivize the adoption of alternative growth techniques. Through the use of convolutional neural networks, this research developed models to accurately classify satellite imagery of rice fields according to their levelling technique. It is our hope this research will be both expanded upon in the field of machine learning and used to effectively focus sustainable farming efforts.

Acknowledgements

I would like to take this opportunity to acknowledge a few individuals who were instrumental to the development and success of this research. The following people each made significant contributions to this study. I am appreciative of all the assistance I received in developing, designing, and conducting this research, and hope this effectively conveys my gratitude to all parties involved. Thank you for your time and effort.

- Dr. Kelly Sullivan — Research Advisor
- Dr. Benjamin Runkle — Field Expert
- Dr. Liang Lu — Field Expert
- Dr. Michele Reba — Field Expert
- Dr. Chase Rainwater — Research Assistance
- Bishwa Sapkota — Software Expert

Table of Contents

1. Introduction

The research conducted in this study aims to be a stepping stone for the agricultural industry in making the transition to a more-sustainable approach to growing food crops. This move is necessary, as critical resources—principally water and the environment—become increasingly scarce. Rice production consumes an incredible amount of water, and in some regions, consumes it at an unsustainable rate. A number of techniques have been developed to reduce the environmental footprint of rice farming (e.g. Alternative Wetting and Drying, De-Nitrification-DeComposition, Biological Nitrogen Fixation, saturated soil culture, etc.). It is easier to implement these techniques into precision-leveled rice fields. However, there is currently no database classifying fields according to leveling technique. The theory behind this study is if such a database were to exist, effective programs could be developed to incentivize the adoption of these alternative growth techniques.

The research performed in this study employs machine learning image classification techniques to classify satellite images of different rice fields according to the leveling technique used on a particular field. Through the use of neural networks, our program was able to classify fields as a contour-levee field or a precision-leveled field. The model was able to classify field imagery with an average of 88% accuracy using a set of 200 images (140 used for training, 40 used for validation, and 20 for testing the model). However, there is much room for expansion and further development of the research presented in the remainder of this paper.

1.1. Agricultural Industry Background

Agriculture sustains our world. Food crops grown throughout the world are vital to sustaining life on this planet. It is no secret the agricultural industry is riddled with uncertainty—irregular weather patterns, droughts, and pests all can be detrimental to a farm's crop yield. A staggering amount of resources—both human and material—go into the different processes used to overcome these obstacles and prepare fields for the planting, growth, maintenance, harvest, preservation, and transportation of

crops. One of the most frequently used resources is water. Water scarcity is a harsh reality that is impacting many regions of the world. This scarcity is predicted to get worse [1].

Rice consumes more water throughout its production cycle than most other crops. However, it has been and remains a principle component of a majority of the world's diet. The state of Arkansas produces over half of the United States' total rice crop [2] and is the largest rice-producing state by volume in the country [3]. The concerns about groundwater depletion within the state are ever increasing. Current water usage levels exceed recharge; the process is unsustainable [2]. However, estimates show annual crop yield is going to have to double by the year 2050 in order to keep up with global demand [4]. Arkansas will be expected to carry some of this load, especially in terms of rice production.

A natural cost that comes with the consumption of such an incredible amount of resources is the negative effects these processes have on our environment. Greenhouse gas emissions are a major concern for people in industry, politicians, and the general population. Methane (CH_4) and Nitrous Oxide (N_2O) are two of the main emissions resulting from agricultural processes. Globally, agricultural soils are responsible for 46-52% of the N_2O emissions [5]. Flooded rice production accounts for 11% of the national methane emissions, making it the third largest emissions source in the United States [6].

Another environmental concern associated with rice production is its high level of water consumption, which results because rice fields traditionally remain flooded throughout the growth process. A 2004 survey conducted by the Arkansas Natural Resources Commission (ARNC) estimated groundwater withdrawals in the state are approximately 24.6 billion liters per year. This number is a 70% increase from 1985 and over 12 times the amount used in 1945 [7]. The rate of increase shown here is alarming.

In order to overcome the looming obstacles of rising production requirements and environmental risks, new methodologies must continue to be explored. The agricultural industry is pursuing sustainable intensification [8]. In rice production, there are several initiatives of this sort; a frontrunner is Alternative Wetting and Drying (AWD). AWD irrigation of rice is a methodology allowing farmers to drain rice fields intermittently throughout the growth cycle rather than having the fields continuously flooded. This

practice has shown remarkable improvements in water usage and emission reductions. AWD has the potential to reduce total water consumption in the rice production cycle by 20-70% with minimal effects on production yield [9] [10]. This methodology also has a positive impact on emissions. The Intergovernmental Panel on Climate Change stated that AWD can reduce CH_4 emissions by an average of 40% in single acreage events and by 48% in multiple acreage scenarios [11]. Methods such as this are a step in the right direction as humanity strives to sustainably intensify crop production to combat global food demands. If Arkansas were to adopt this technique at the state level, water usage efficiency would greatly increase while emissions would show a significant decrease. There are ongoing field experiments being conducted at the University of Arkansas' Rice Research and Extension Center near Stuttgart, AR examining the effectiveness of alternative rice production techniques [12]. The implementation of a process such as AWD—one involving multiple cycles of draining and flooding rice fields—is easier to execute in a precision-leveled field. Field maintenance and irrigation are easier to manage in this case. It is reasonable to conclude AWD would then be both more sustainable and more successful in fields or regions of fields leveled with high precision.

1.2. Motivation

Rice fields are traditionally leveled in one of three ways: zero-grade, precision-level, or contour levee. A zero-grade field experiences no elevation change throughout the field. Precision-leveled fields use straight levees and a controlled gradient from one end of the field to the other. Contour-levee fields are ungraded fields using levees to create small water basins of equal elevation but different sizes to improve the above-ground water storage. Figure 1 shows the visual difference between a zero-grade/precision-leveled field (outlined in blue) and a contour levee leveled field (outlined in red). A 2002 survey revealed 55% of fields within the state of Arkansas are irrigated with contour levees, and that 45% of rice fields are precision-leveled. The survey also stated 5% of the precision-leveled fields were zero-grade [13]. This survey is out of date and fields are continually changing. Precision-leveled fields

improve irrigation and machine efficiency and positively impact yields by an average of 10%. Zero-grade rice fields have shown water usage reductions of 60% [14]. Although the capital investment to have a field professionally leveled is high, subsidy programs are in place and the positive impact it can have on the farm's production yields and carbon footprint is significant.

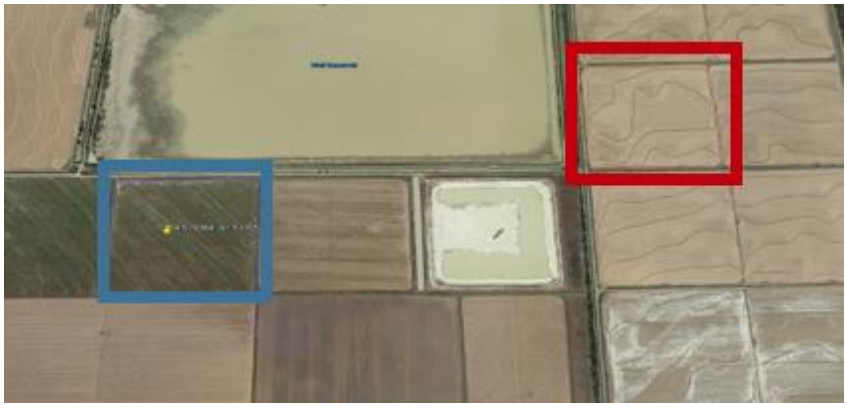

Figure 1. The visual differences between rice field leveling techniques. Contained in the red square is a contour-levee leveled field, while the blue square shows a precision-leveled field.

Implementing new irrigation techniques and increasing the amount of precision-leveled fields used in the rice industry are two valid strategies for sustainably increasing production. At present, there has been relatively little work done to promote the integration of these two strategies. Particularly, there is not currently a database of rice fields categorized by leveling technique. A higher degree of precision in field leveling results in significant efficiency improvements for the farm's operation. Irrigation costs are reduced significantly in a precision-leveled field by lowering the total water volume needed. An increase in machine efficiency comes in the form of reduced cost and usage during planting and harvesting [13]. The difficulty inherent in the process of draining a rice field is also significantly reduced with an increase in the precision of the leveling technique used in a particular field. It is imperative that a precision-leveled field be well-drained in order to remain profitable [13]. The theory behind the process of AWD involves multiple draining periods throughout one growth cycle. A field possessing a more-precise leveling

technique would provide a much simpler, and therefore a more successful, implementation and management process of the AWD growth methodology.

Alternative Wetting and Drying has already shown promising results in improving farm efficiency in its pilot experiments conducted on U.S. soil. A 2014 study conducted near Stuttgart, AR used three different AWD water consumption levels in an examination of the process compared to traditional flooding. This study showed a water reduction of 18-44% for the three levels vs. traditional flooding techniques [15]. A similar study conducted in 2015 (also near Stuttgart, AR) continued to show water use efficiency improvements: a 21-57% increase in efficiency [2]. These efficiency figures show strong potential to positively impact the currently-unsustainable consumption rate of water from the Alluvial Aquifer (the aquifer used by Arkansas Delta region rice farms).

The correlation between precise field leveling techniques, the success of sustainable growth methodologies like AWD, and the overall improvement of farm efficiency measures has become clear. If an up to date catalog of rice field leveling techniques were to be created, the process of implementing, testing, and monitoring the sustainable production of rice in years to come would be notably expedited. Machine learning and image recognition software programs are capable of discerning rice fields into the appropriate leveling category. It is possible to use these programs, given a set of pre-classified images the program can use to learn the differences between fields, to dynamically classify current rice field leveling techniques for larger areas. By dynamic we mean the program would be capable of automatically identifying fields' leveling classifications as new satellite imagery becomes available. This research will help to promote precision-leveling techniques as well as alternative irrigation methods like AWD, and it will create positive publicity for the creation of incentives and government-funded carbon credit programs in the agricultural industry. These programs would provide financial incentives to urge farmers into adopting the new processes as soon as possible, which will be a vital stepping stone towards a sustainable rice industry.

2. Machine Learning Review

Machine learning is a tool capable of addressing the challenges outlined in the previous section. A solution to overcoming these challenges can be found through machine-learning based image classification. In recent years, machine learning technology has made great strides in the field of image classification. Today's software possess the ability to categorize images based on attributes contained within the images themselves. By training a model using manually-labelled images, the program can then be applied to a much larger set of field images in order to produce a functioning database for the leveling technique of rice fields. Code can be written in order to perform this classification analysis. This code can then be used to classify fields at a much larger scale—potentially state- or nation-wide. This database would serve as a springboard for farmers towards a more sustainable age of agriculture. The remainder of this section overviews the machine learning discipline, and section 3 describes how machine learning is applied to field imaging. Sections 4, 5, and 6 discuss the experimentation, results, and conclusions and future work associated with this study.

2.1 Machine Learning Overview

Machine learning is a subfield of computer science that gives computers the ability to learn without being explicitly programmed [16]. This allows computers to automatically improve performance at some task through experience [17]. Machine learning techniques can help by automating time-consuming processes in knowledge acquisition critical to developing a knowledge-based system [17]. This is one of the driving reasons why machine learning techniques were employed in this research study. This task is not feasible to perform by hand at a grand scale. It is simply too great an amount of work, not to mention the possibility of a field's leveling technique changing with time. Further, machine learning is useful in cases where algorithmic solutions are not available—there is not a formal model about the application considered in this study [18]. By this, we mean there is not an algorithm having all of the answers. There is just simply too much variance in fields for this algorithm to exist with today's

technology. However, machine learning provides sophisticated estimation techniques for scenarios such as this. Additionally, the use of machine learning automation techniques reduces the cost of development by decreasing the time needed with experts and knowledgeable engineers. These techniques can also uncover details that might be overlooked by experts [17].

The work presented in this paper focuses on the image classification portion of machine learning. Image classification has a wide range of applications including manufacturing, telecommunications, marketing, and scientific analysis [17]. Because this is a developing field, there are many applications of this technology yet to be uncovered. A wide variety of techniques can be applied to these types of problems. Figure 2 summarizes the existing techniques used for working on machine learning problems.

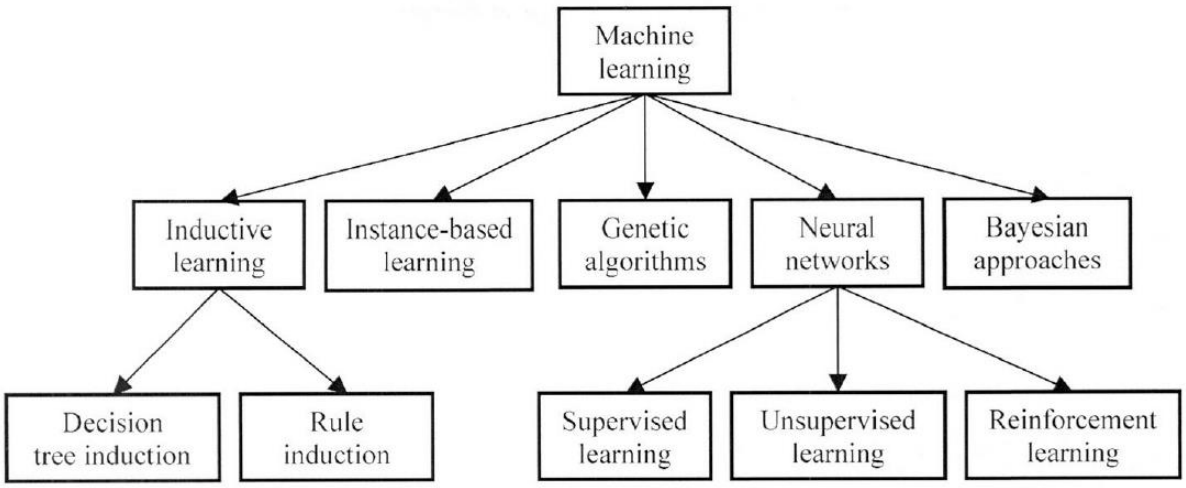

Figure 2. A classification of the main machine learning techniques *[17]*

In general, image classification may be broken down into a series of steps. A survey conducted by Kamavisdar in 2013 breaks the process down into four steps: pre-processing, detection and extraction of an object, training, and classification of the object [19]. Lu expands these steps further: determination of a suitable classification system, selection of training samples, image preprocessing, feature extraction, selection of suitable classification approaches, post-classification processing, and accuracy assessment [20]. Both Kamavisdar and Lu provide detailed summaries of the different image classification techniques

available for use [19] [20]. A review of classification techniques written by Kotsiantis shows the following flow diagram for the general process behind image classification [21].

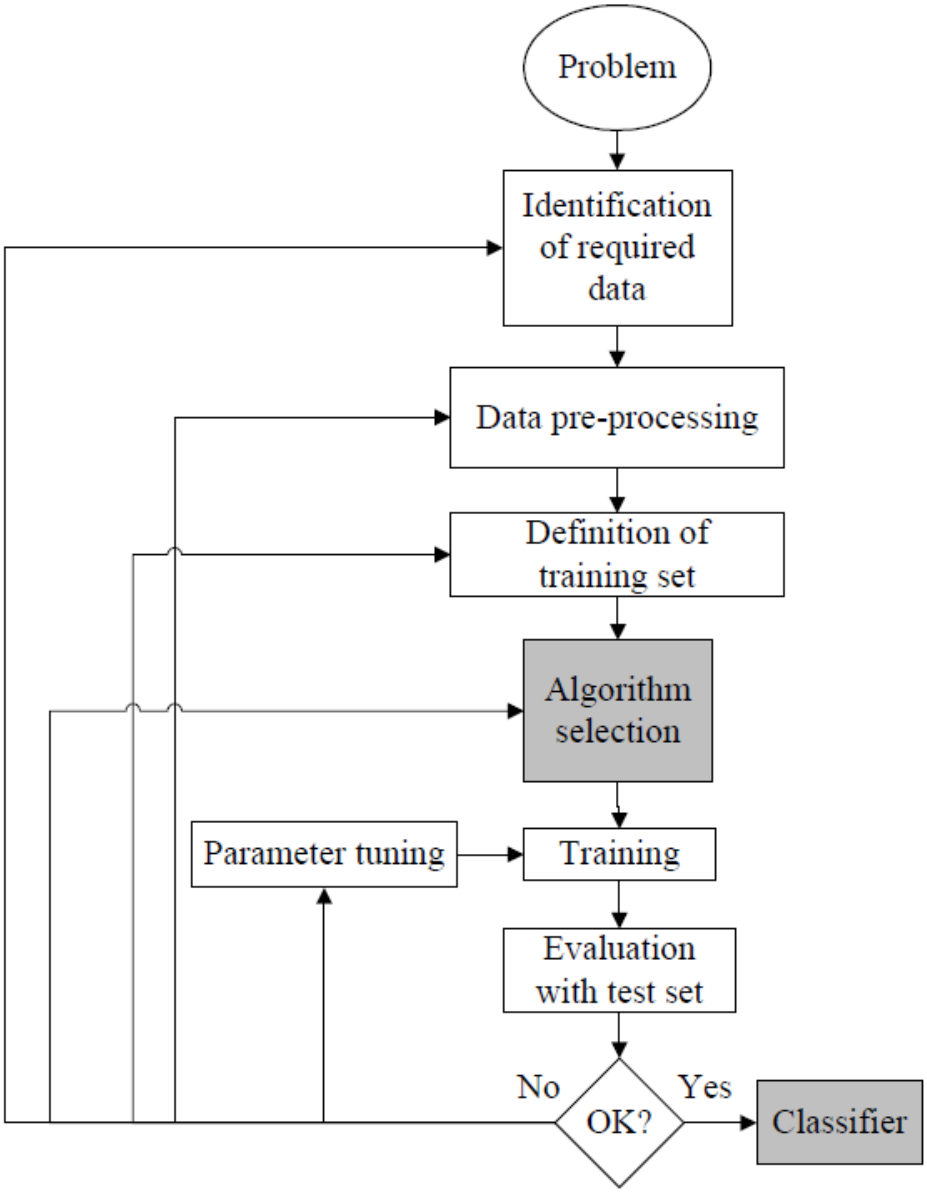

Figure 3. Process flow diagram for image classification *[21]*

Image classification is divided into two broad categories: unsupervised learning and supervised learning. In unsupervised learning, a target function for prediction has not been established. The goal in

this case is to group or cluster instances based on some similarity or distance measure. In supervised learning, however, there is either a nominal or continuous-valued target function to be predicted [17]. Generally, a supervised learning problem must predict the labels of patterns. These labels can be a class label or a real number [22].

Models performing image classification have three broad steps: training, validation, and testing. During the training section, the model receives partial information about the true relationship between patterns and their labels through a number of correctly labelled instances [22]. The training portion of the model is what the computer uses to learn about the dataset. A number of advantages are realized from training models in this way: it can save an enormous amount of effort in design, and it can be used for problems that cannot be easily specified precisely in advance, perhaps because the environment is changing [22]. This is especially applicable to the work done in this study, as the leveling technique used in rice fields may change from year to year.

The validation and test data sets are also critical components of any image classification experiment. These records may be collected in a number of different ways. However, a common practice is to split the complete set into thirds, using two thirds of the images for training, and the remaining third for validation and testing [21]. Usually, the validation set is roughly twice the size of the testing set. The validation set is used to validate the parameters and predictions the model learned in the training set. Then, the test set is used as a way to evaluate the model's classification accuracy. It is important to randomly assign which images are partitioned into each data set with each iteration of the model [23].

Three key questions must be considered when designing a supervised learning problem experiment: approximation, estimation, and computational efficiency. The approximation concern centers on the question of whether or not the model is strong enough to identify the true relationship between patterns and labels [22]. The estimation concern stems from the question of whether or not the amount of data given to the model suffices to represent the relationship with a desired source of accuracy. Ideally, the data set would be infinite to allow the model to train indefinitely. However, in all practical scenarios, the training set will be finite [22]. The final concern addresses the model's computational efficiency. This

concern brings to light the question of how to efficiently make use of the training data to determine an accurate representation of the true relationship [22].

Choosing which type of image classification model to use for the purposes of this research study mandated the consideration of these three principle concerns. Ultimately, a neural network, supervised-learning approach was chosen. There are a number of advantages to using neural networks for image classification. Lu lists the following advantages: its non-parametric nature, arbitrary decision boundary capability, easy adaptation to different types of data and input structures, fuzzy output values, and generalization for use with multiple images. This makes it a promising technique for land-cover classification [20]. The flexibility and robust nature of neural network models and the advantages provided in land-cover classification scenarios were driving factors behind the decision to use a neural network architecture for this study.

2.2 Neural Network Overview

An easy-to-understand example of an artificial neural network is the simple perceptron. At their most essential level, these are binary models—the outputs from a simple perceptron model predict an input to be a member of one of two predefined classes. The goal of the perceptron model is to compute appropriate values for the parameters defining the decision hyperplane [24]. The decision hyperplane assumes the two classes in the model are linearly separable. Plainly, this model works to define a linear function dividing the two classes such that no misclassifications are experienced. The model can be expressed in terms of parameters $\omega = (\omega_1 \ldots \omega_n)$ and $\theta \in R$, as

$$f(x) = \text{sgn}(\omega * x - \theta), \qquad \text{(Eq. 1)}$$

where x is an n-vector of data and sgn(y) equals 1 if y > 0 and 0 otherwise [24].

The model begins with arbitrary values for its parameters. As the training data is processed, the parameter values are updated upon the misclassification of an example [22]. In the case of a misclassification, a prescribed, fixed, positive constant is used to scale the model parameters. By

adjusting the parameters, the decision hyperplane shifts towards the misclassified point [24]. Essentially, each time the model incorrectly predicts a label for a pattern, the model adjusts its parameter values to improve its accuracy for future predictions. This makes sense; a pattern incorrectly labelled is one that resides close to the middle ground—or grey area—between being classified as 0 or 1. When the model begins to encounter these instances, its decision boundary migrates towards them. As the model continues to train, it gets closer and closer to the 'true' decision boundary [22]. To help illustrate this example, the following two figures have been included. Figure 4 below shows the arbitrary assignment of the model parameters ω and θ. The dashed line represents the decision boundary. Figure 5 shows a point '*x*'. In this case the point is misclassified; the parameters ω and θ are updated, moving the decision boundary closer to '*x*' until the decision hyperplane correctly separates the two classes [24].

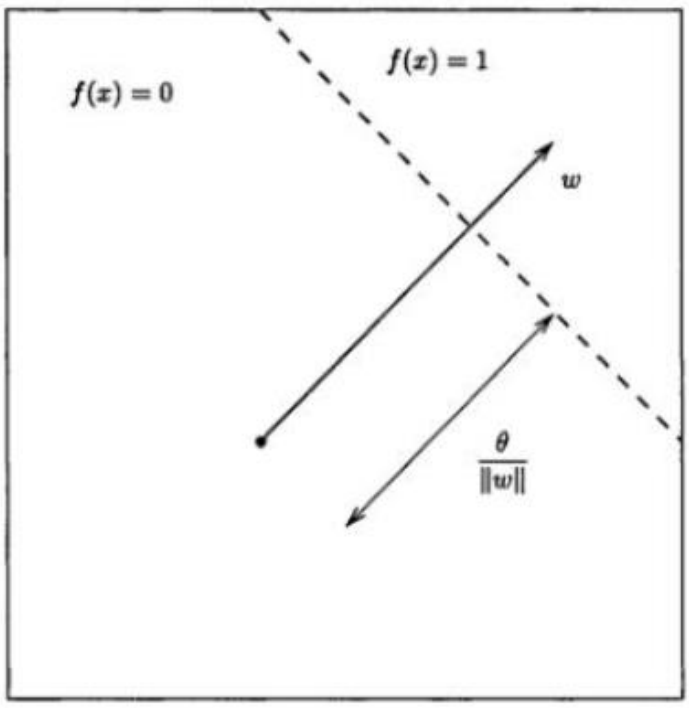

Figure 4. The initialization of a simple perceptron model *[22]*

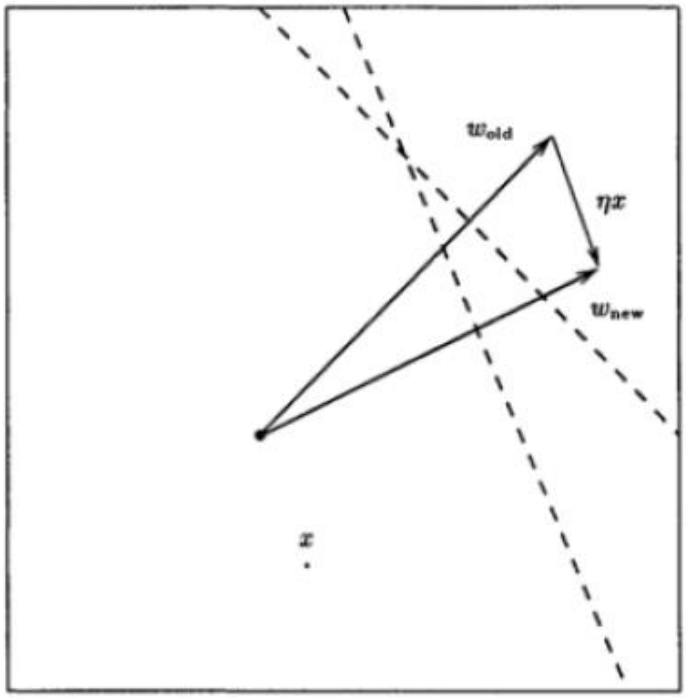

Figure 5. Simple Perceptron Model Parameter Update and Decision Boundary Translation *[22]*

In the case there are multiple perceptrons considered in one model—a much more practical scenario—the model iterates through all inputs until it finds an optimal solution. The perceptron algorithm examines all perceptrons, updating the associated parameter values ω and θ until the appropriate values needed to linearly separate the two classes with the decision hyperplane are determined [24]. Neural networks generalize perceptrons in several ways: (1) nodes may be combined into multiple layers such that the output from one or more nodes is used as input to another trainable node. (2) Each node may use a more general (e.g. nonlinear) model to transform input into output. (3) Each node's output need not be binary. For an in-depth discussion on foundations in artificial neural networks, see Martin's book *Neural Network Learning: Theoretical Foundations* [22] or *Pattern Recognition* by Theodoridis, et al. [24].

3. Methods

3.1 Convolutional Neural Networks

As discussed above, the research in this paper uses neural network technology for the purposes of image classification. Specifically, this research employs the VGG16 model (which can be fine-tuned using our rice field data) developed by the Visual Geometry Group at the University of Oxford. The group designed this convolutional neural network to help them win the 2014 ImageNet challenge. Through the development of this model, the team effectively redefined the 'state-of-the-art' in image classification accuracy. Although the VGG16 model has been beaten since it was released in 2014, it is still a high-quality model for the purposes of image classification. ImageNet has played a vital role in the advancement of image classification and deep visual recognition architectures. By way of competition, each year teams push the boundaries to new heights, developing stronger, more accurate neural networks. For the research presented in this paper, however, we credit the Visual Geometry Group for the design of VGG16 [25].

VGG16 is a convolutional neural network. A convolutional neural network (CNN) is quite similar to a regular neural network. The main difference is CNNs make the explicit assumption that provided data are images. CNNs generally make strong assumptions about the nature of images, allowing the network's architecture to better model image properties [26]. Because these properties are included, the model becomes more efficient at using the feed forward functions, vastly reducing the number of parameters contained in the network. Generally, CNNs contain convolutional layers, pooling layers, and fully-connected layers (also present in standard neural networks) in their architectures [27].

Each of these layers serves a critical purpose. The two-dimensional pixels of each image fed to the network serve as data for the model. Within the convolutional layers, these images are convolved with multiple learned kernels using shared weights [28]. In the pooling layer, the size of the image is reduced while maintaining its information using max-pooling. These two layers make up the feature extraction portion of the model. Next, the extracted features are weighted and combined in the fully-connected

layers. These fully-connected layers are similar to the layers in a standard multilayer perceptron model [28]. The final layer uses softmax regression (multinomial logistic regression) to identify the strongest output neuron for each classification. Each neuron's output represents the probability an image belongs to a particular class [28]. Figure 6 below provides a visual representation of a general CNN architecture.

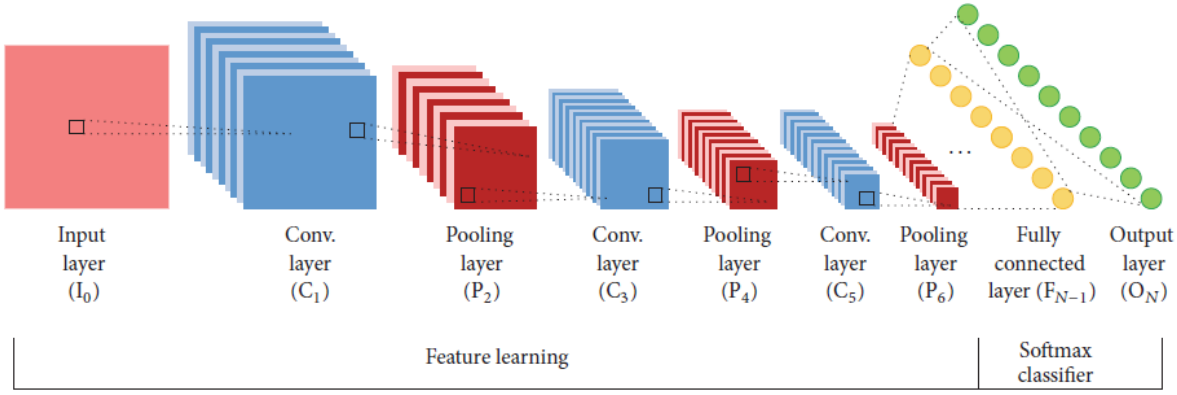

Figure 6. A typical convolutional network architecture *[28]*

The VGG16 model is a pre-trained model. It is common for CNNs to be trained using some variation of the gradient-based backpropagation method [28]. VGG16 was trained by optimizing the softmax regression objective based on backpropagation techniques [25]. Training patterns and expected outputs are fed into the network. The network error is then backpropagated through the network to compute the gradient of the network error with respect to the weights. This gradient is then used to update the weight values according to a specific rule [28]. In the case of VGG16, the rule used was momentum [25]. This process is similar to updating the parameters in a multilayer perceptron model to shift the decision hyperplane towards an optimal value.

The height, width, and depth dimensions are important in CNNs. The height and width refer to the dimensions (pixel size) of the image being fed into the network. The depth represents the number of color channels considered by the network [27]. Depth is the factor the Visual Geometry Group chose to focus their efforts around when designing VGG16, named after its sixteen layers. The depth of a CNN is a

critical aspect of design [25]. In designing VGG16, the group chose to essentially fix other parameters of the architecture, and increase the depth of the network through the addition of more convolutional layers. This is feasible because of the use of very small convolution filters in all the layers (3x3 pixels) [25]. The model's sixteen layers are made up of convolutional layers, fully-connected layers, and a final layer to perform softmax regression. For more technical information about the network's architecture and the theory behind its design, see Simonyan's paper *Very Deep Convolutional Networks for Large-Scale Image Recognition* [25].

CNNs are a sophisticated machine learning tool used to help us understand image-based data. But what do CNNs actually 'understand' about these images? According to "The Keras Blog", an online blogging forum, CNNs really understand two things: 1. a decomposition of their visual input space as a hierarchical-modular network of convolution filters, and 2. a probabilistic mapping between certain combinations of filters and a set of arbitrary labels [29]. What does that mean? VGG16 uses multiple image filters to detect significant patterns in images. The first few layers encode direction and color. These filters are combined into basic grid and spot textures. These patterns are typically combined into increasingly complex shapes. As more and more layers are stacked on top of each other, intricate patterns are developed which are used for classifying patterns occurring in the actual images being fed to the network [29]. Figures 7-9 show the general development of filters within VGG16 as the model progresses through the different layers.

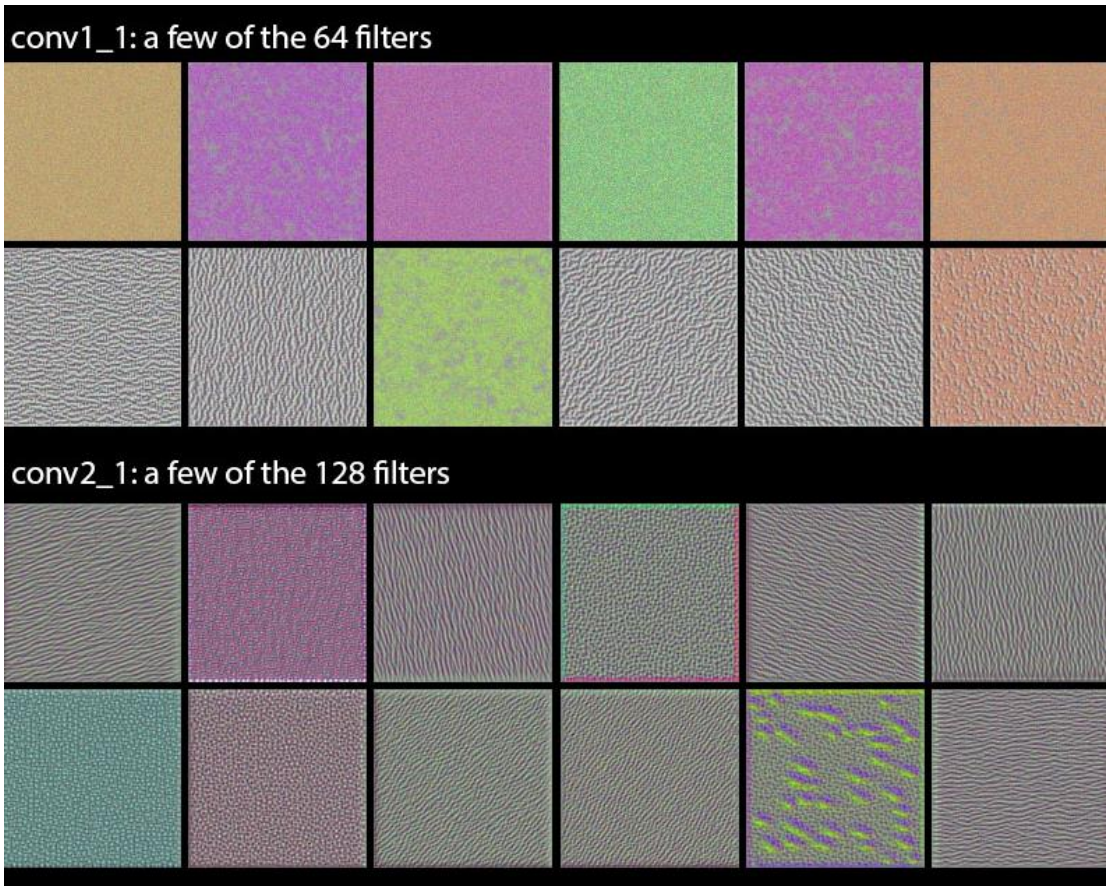

Figure 7. VGG16 convolutional layers 1 and 2 *[29]*

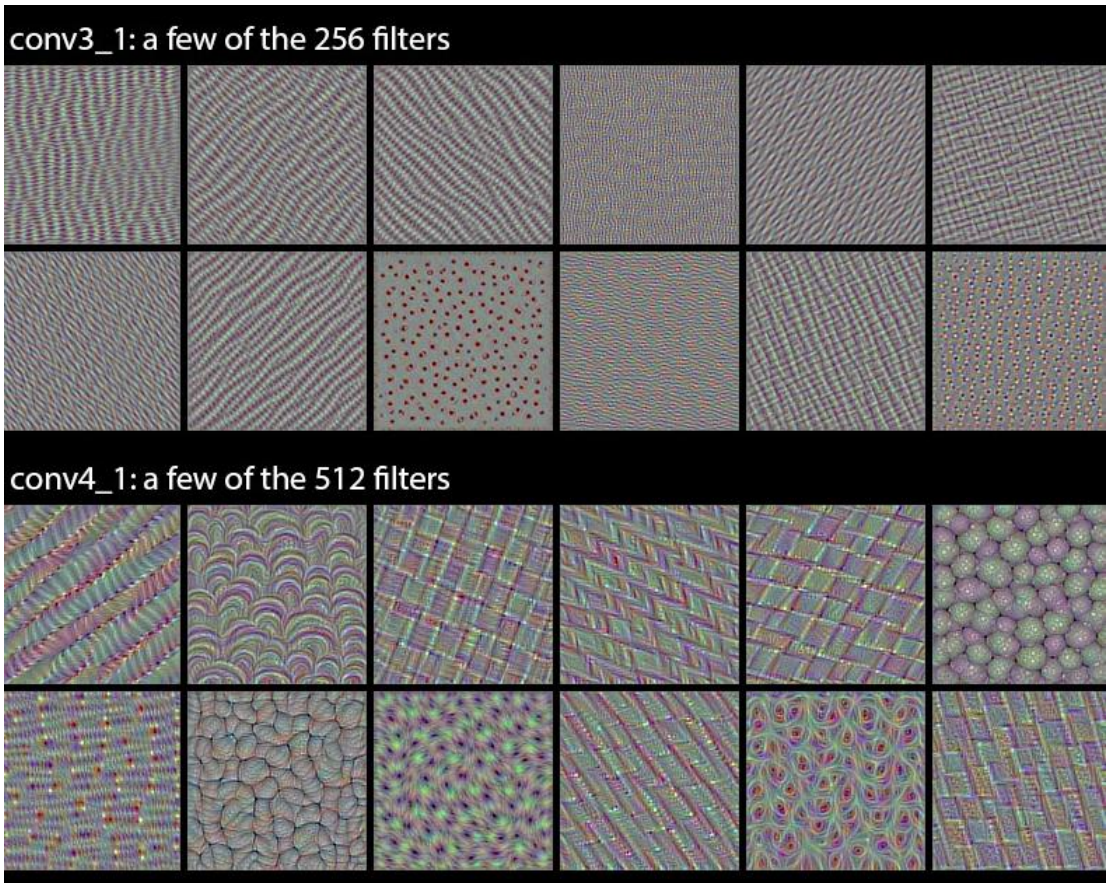

Figure 8. VGG16 convolutional layers 3 and 4 *[29]*

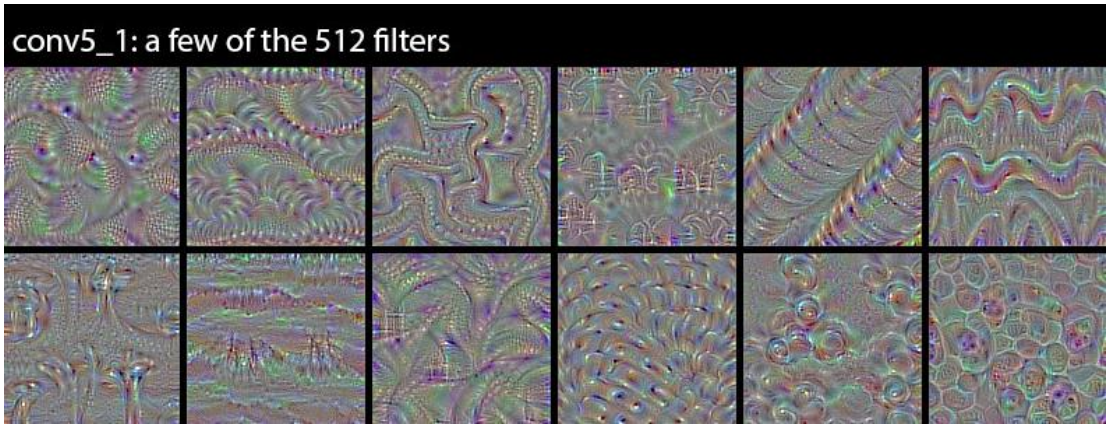

Figure 9. VGG16 convolutional layer 5 *[29]*

3.2 Hardware and Software

The VGG16 model was essential to the success of this research study. However, it was not the only component. The processing power needed to handle the large number of images necessary to effectively train a CNN requires the use of a GPU. The environment used to execute the experiments in this study was modeled directly after those used in the FastAI course "Practical Deep Learning for Coders: Part 1." This coursework was used as a tutorial for learning the concepts associated with deep learning image classification. FastAI teaches these concepts through the use of Amazon Web Services. One GPU server was leased from AWS for the purposes of completing this project. In order to properly execute the VGG16 model, multiple python libraries were used including: Keras, Theano, numpy, utils, os, sys, vgg, PIL, and sklearn, among others. An alternative library to Theano for image classification is TensorFlow. Essentially both of these libraries would perform the same service for this model. However, the TensorFlow library is rather new. Therefore, the amount of information and support available online for diagnosing errors is limited. Further, Theano is actually somewhat faster with convolutions in GPUs than TensorFlow is at this time [29]. For these reasons, Theano was chosen for our project.

The deep learning code used to classify the rice field images into two categories—'Levee' or 'Zero' in our code's case—is modeled after the 'Dogs vs. Cats Redux' notebook in the FastAI course files [23]. There were significant changes made to the code to permit execution using a different terminal environment (Secure Shell Client). The techniques used to access different files and folders in our directory structure were adjusted to mesh with Secure Shell Client. Further, modifications were made to the code so that multiple iterations of the experiment could be executed in one pass from the command line.

3.3 Model Evaluation

The optimization of model parameters occurs as a result of iterating the experiment over a wide range of values for important variables contained within the model. The set of parameters which produced the best results were recorded and may be used for further work on this problem. It is important to note the optimal parameter values for this model may be different when performing image classification on a different set of images. What measures determine how well the model performed? The model used in this study recorded two common measures for model evaluation: accuracy and log loss.

Accuracy looks at the number of images the model labelled correctly. It is a count of the predictions made equal to their true value [23]. Though this can be helpful in seeing how well the model is performing, it is not always the best way to assess a model's performance. Because of its discrete nature (i.e. a predictor is either correct or incorrect), accuracy does not take into account how confident the model was in making any particular prediction [23]. For example, imagine a model differentiating between pictures of contour-levee fields and precision-leveled fields. Take a picture of a contour-levee field. If the model decided it was 49% sure that picture displayed a contour-levee field, it would say that picture showed a precision-leveled field. Conversely, if it were to be 51% positive the picture showed a contour-levee, it would guess so. The model isn't very confident in either scenario; this is a grey area. Only using accuracy neither rewards the model for being confident in some predictions nor reprimands it for being wary about others. In an extreme example, let's say the model was 99% sure an image showed a picture of a contour-levee field. If that image was a precision-leveled field, accuracy is not severely affected even though this prediction was way off.

Log loss becomes valuable in this situation. Log loss does, in fact, take into consideration the uncertainty associated with the model's predictions. Again, take a picture of a contour-levee field. As the model more confidently declares this image to be a contour-levee (i.e. as the confidence in that prediction grows), the log loss associated with that prediction decreases. On the other hand, as the model becomes less confident the image displays a contour-levee field (i.e. as the confidence in that prediction decreases),

log loss increases rapidly. Effectively, log loss addresses both type one and type two errors, but more significantly penalizes those predictions which are confident and wrong [23]. The figure below shows a graph of the log loss function when the true label value is equal to one.

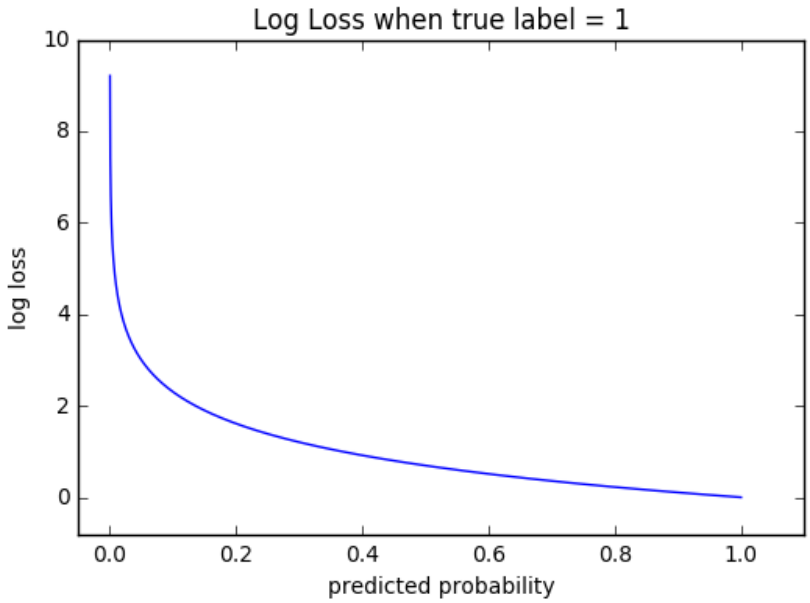

Figure 10. Log loss function *[23]*

These measures were used to evaluate the different input values passed into the model from the command line. As the model progressed through the training set, it continued to adjust its parameters until the optimal values were determined for classifying images of rice fields.

In order to best explain these two evaluation measures, it is important to explain how the model records them. The optimization algorithm used to select model parameters from training data is called gradient descent. Generally, this algorithm is used to minimize some function by iteratively moving in the direction of steepest descent. The relative steepness of descent in any direction is determined by the negative of the gradient [23]. The figure below and the following scenario are used by FastAI to explain the concept of gradient descent. Taking the map below in the context of a cost function, the goal is to find the best route from the top of the mountain to the bottom of the sea. The arrows represent the direction of steepest descent—negative gradient—from any given point. If the path of travel follows the path of

steepest descent, the cost function is reduced at the fastest possible rate. The direction of the steepest gradient is calculated with every step, showing the model which way to go next [23]. The size of the 'step' taken by the model is called the learning rate. A small learning rate will take smaller steps, but may become computationally expensive. Conversely, a larger learning rate will take big steps, possibly overstepping local minimum values. This process is repeated until a minimum is found.

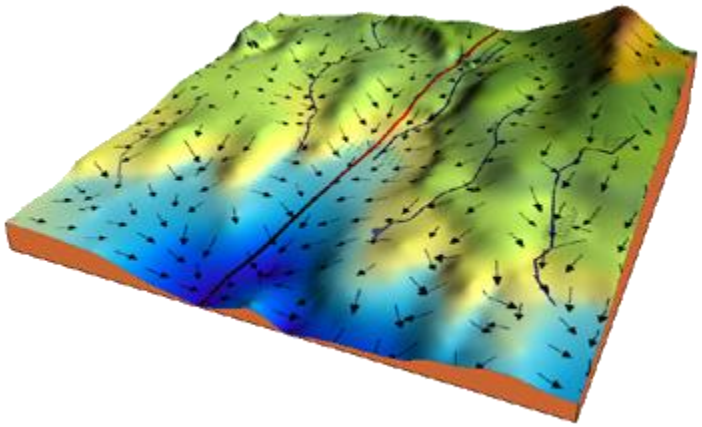

Figure 11. Topographical map explaining the concept of gradient descent *[23]*

Developing a dataset of rice field images was critical to the success of this research project. The following section describes the images and the process used to acquire them.

3.4 Data

The data used to carry out this research study are satellite images of rice fields. The images are all of fields from the Arkansas Delta region, generally around the city of Stuttgart, AR. Because of the large presence of rice around this community, we chose to concentrate our data collection efforts here.

Image quality is a major concern for people working on similar problems to the one examined in this study. The importance of having high-resolution images of fields when performing work like this in the agriculture sector is high, as it can have a significant impact on the outcome of experiments. For this reason, the images used as data for the CNN in this study were derived from the National Agricultural

Imagery Program (NAIP) data set. The NAIP data set acquires satellite imagery during growing seasons to make available for government and public use. This data set has unmatched resolution, having a horizontal accuracy matching within six meters of photo-identifiable ground control points used during image inspection [30]. To access this imagery, a personal geodatabase was created in the ArcGIS ArcMap 10.3.1 software environment. With the background imagery of Stuttgart, AR and the surrounding area loaded into ArcMap, a series of steps were repeated to obtain the .JPG images at a field level. These steps may be found in Appendix 1. Once a .JPG file had been exported from ArcMap and stored in the appropriate directory, the process was repeated.

In this study, the process described in Appendix 1 was repeated to obtain 200 images of rice fields in and around the Stuttgart, AR area. These pictures were uploaded into a backup directory on the AWS GPU discussed earlier in section 3.2. In order for the python program to properly execute in this study, it was important to name the .JPG files according to their classification. In this study, the name of any image file was either ‘Levee#.JPG’ or ‘Zero#.JPG’. This is critical because the beginning of the python program will copy all of the contents from this backup directory on the AWS GPU server and paste them into their respective classification directories (Levee or Zero in this case) based on the file name. This initial separation of files is vital to the success of the CNN. Without the successful population of the training directories from the backup directory, none of the experiments execute properly. An example of each classification’s .JPG image can be seen below.

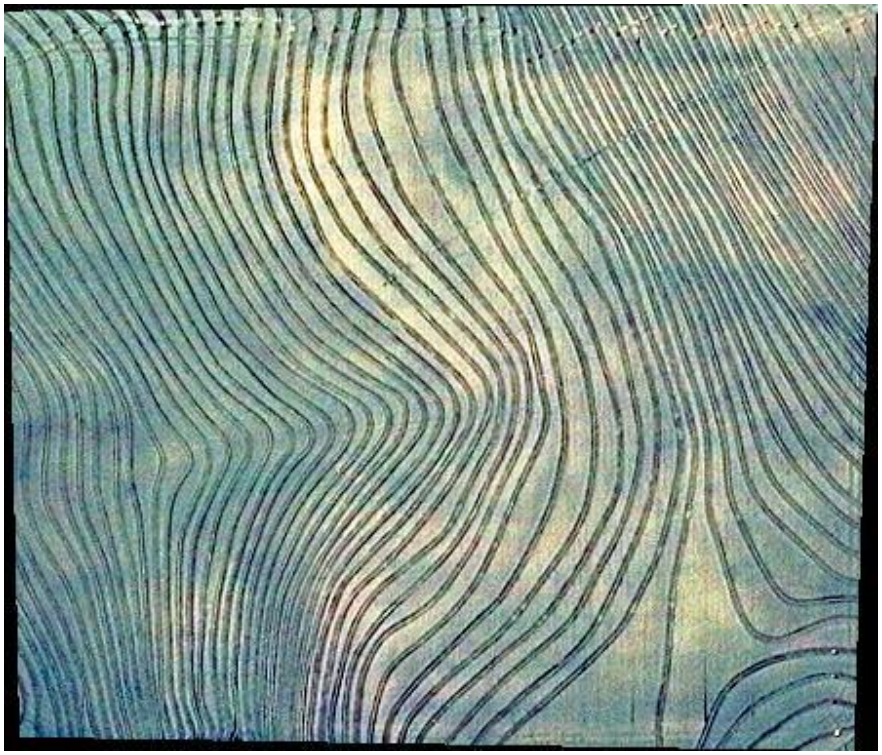

Figure 12. Example .JPG file for a contour-levee levelled field

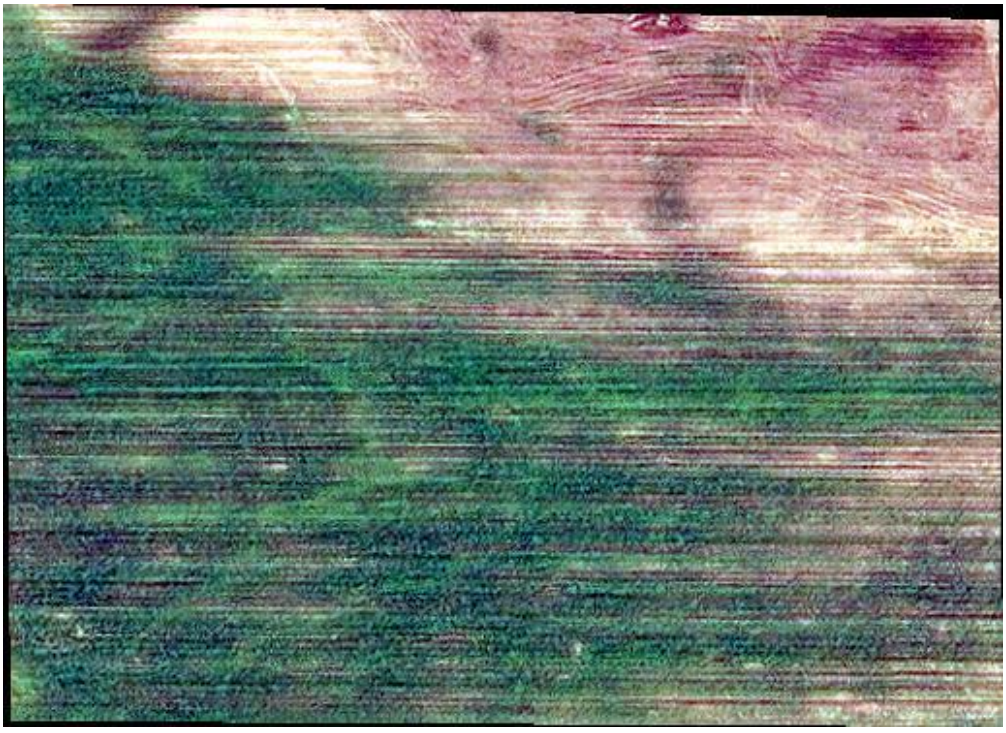

Figure 13. Example .JPG file for a precision-levelled field

4. Experimentation

In order to effectively evaluate model performance, it is important to execute a large number of experiments. By doing so, it is possible to evaluate different combinations of model inputs based on the classification accuracies of each run. Upon examination of the model, we determined there was a small set of input features desirable to pass into the model from the command line. Some of these inputs remained the same throughout our analysis, while others were changed over the iterations. The size of each of the three sets—train, validation, and test—were kept the same throughout the experiments performed in this study. However, the random partition functions included in the code ensured which images were assigned to which set was randomized with each model execution. This input variable was used to separate the validation and test sets from the training set and was set to 60 images. The dataset used in this study contained 200 images. It is common practice to place roughly 70% of images in the training set, while sending the remaining 30% to the validation and test sets [21]. For the purposes of this study, the 30% of images excluded from the training set were again divided into thirds, with two-thirds of them being placed into the validation set and the remaining third in the test set (40 images and 20 images, respectively). The other model input held constant throughout the analysis in this study is the model's learning rate. This input was held to a value of 0.01. This value was chosen because it is a standard used in the FastAI online course on deep learning [23].

Two model inputs were varied over the different experiments performed: the model's batch size and the number of epochs. Batch size refers to the number of images processed at one time by the GPU server. A small batch size may not be computationally demanding enough to occupy the server, while a large batch size may slow down the server's performance. An epoch refers to the number of times the model passes images through the classification algorithm. Each model executes multiple epochs, generally becoming a more-accurate classifier as it progresses. Although the model may benefit from an increased number of epochs used in each iteration, again the tradeoff is in the computational load placed on the server. As the number of epochs increases, the time required to classify the images increases. The

values tested in preliminary analysis for each of these two parameters are shown below. Each combination of batch size and number of epochs underwent five repetitions to account for a poor partition of images. The preliminary analysis executed 150 instances of the model, corresponding to five replications of each combination of Batch Size = {2, 4, 8, 12, 16, 20} and Number of Epochs = {5, 10, 15, 20, 25} to identify which combinations of model inputs produce high classification accuracies and which do not.

The preliminary testing produced interesting results. Essentially, the model either seems to perform well or not well at all without much grey area in between. The measures used to evaluate these experiments are the validation and test accuracies for each iteration. The validation accuracy measures the proportion of images correctly classified in the validation set. This set serves to assess the model parameter weights—a good score here indicates a good selection of parameter weights. The test accuracy statistic measures the proportion of correctly classified images in the test set. Log loss is recorded for each epoch, but not for each overall iteration. For the general application of this research, the two accuracy statistics are sufficient evidence of good or bad classification. The scatter plot below in Figure 14 shows the results of the 150 preliminary iterations. Tables 7-10 in Appendix 2 contain the average accuracy scores and the standard deviations across all five iterations for each model input combination.

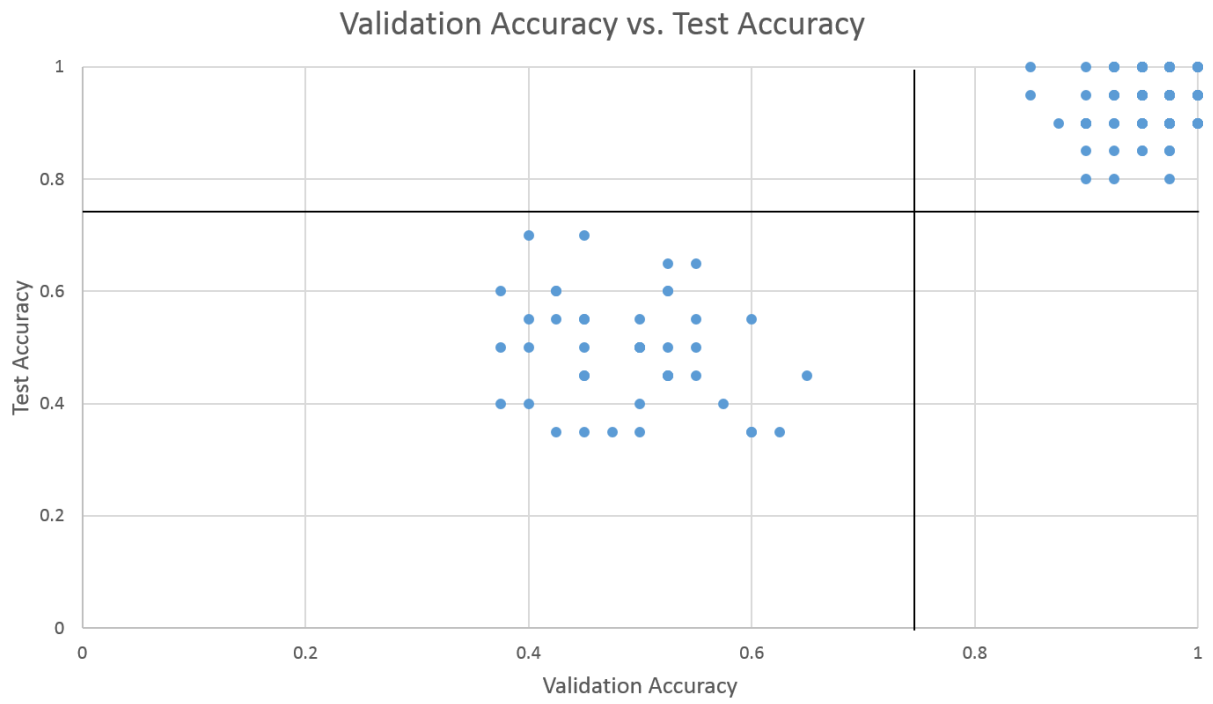

Figure 14. Scatter plot showing the validation accuracy vs. test accuracy from the preliminary analysis

The scatter plot shows an interesting clustering of results over these 150 iterations. It is apparent the model's validation and test accuracies are generally positively correlated. From the scatter plot above, we see if the model's validation accuracy is greater than 80%, the test accuracy will be at least 80%. This is promising, as a strong validation of parameter weights should result in a strong classification of the test set images. Conversely, if the model's validation accuracy falls below 70%, the test accuracy will be at most 70%. Again, this makes sense; poor selection of parameter weights will produce low validation and test accuracy scores. While it appears the low-performance cluster is larger than the high-performance, the number of iterations contained in the high-performance cluster more than doubles that contained in the low-performance cluster. Many results produced overlapping points in this plot. Tables 1-3 below summarize the results from the preliminary analysis. Overall, the model classified field images with an 82% accuracy. However, if the model produces a strong validation accuracy (the high-performance cluster), this accuracy statistic climbs to nearly 95% for field classification. Further, a surface plot of the

different combinations of Validation and Test accuracy was created to show the frequency of the accuracy results. The surface plot and histograms shown below in Figures 15-17 summarize these results. This helps to explain the relative sizes of the clusters shown in Figure 14. The noise in the foreground of the surface plot can be attributed to poor model input value combinations. A full summary of the results of the preliminary analysis experiments may be viewed in Appendix 2.

Table 1. Summary statistics for the entire pool of preliminary analysis runs (both high and low-performance clusters)

Statistic	Validation Accuracy	Test Accuracy
Count	150	150
Max	1	1
Min	0.375	0.35
Average	0.823833333	0.816333333

Table 2. Summary statistics for the low-performance cluster in preliminary analysis

Statistic	Validation Accuracy	Test Accuracy
Count	43	43
Max	0.6500	0.7000
Min	0.3750	0.3500
Avg	0.4901	0.4942

Table 3. Summary statistics for the high-performance cluster in preliminary analysis

Statistic	Validation Accuracy	Test Accuracy
Count	107	107
Max	1.0000	1.0000
Min	0.8500	0.8000
Avg	0.9579	0.9458

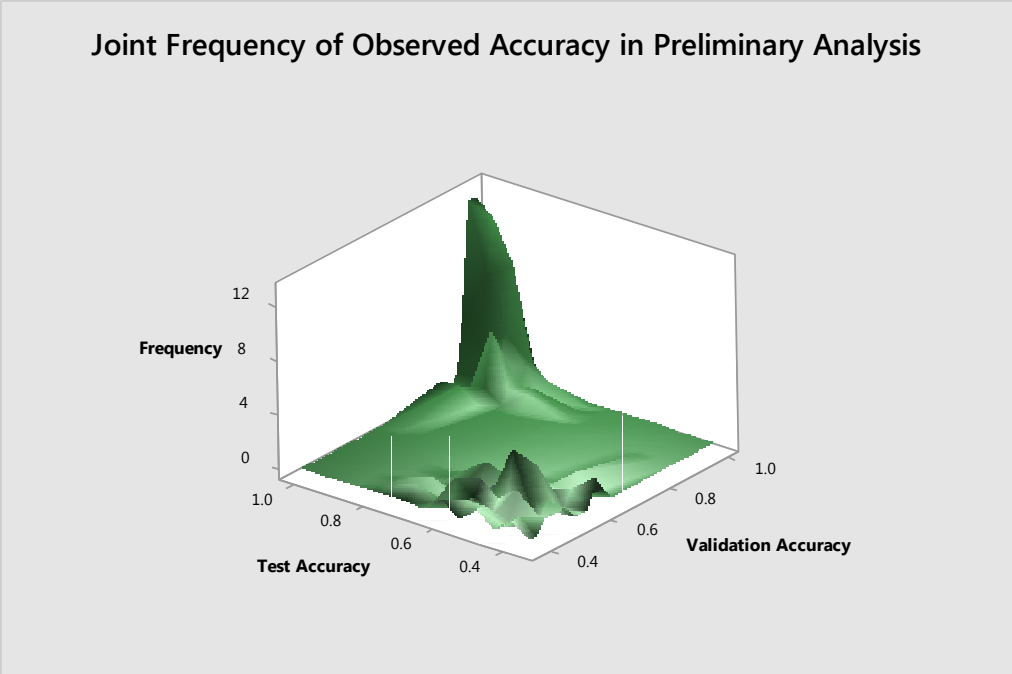

Figure 15. Surface plot showing the frequency of the different pairs of accuracy results for the validation and test sets: preliminary analysis

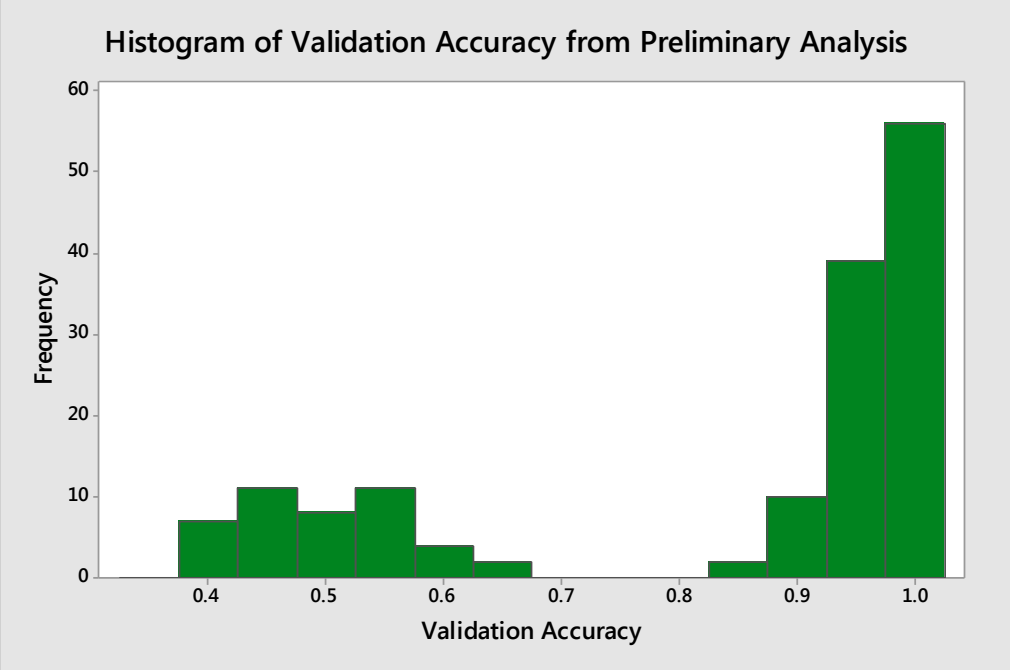

Figure 16. Histogram showing the frequency of accuracy results for the validation set: preliminary analysis

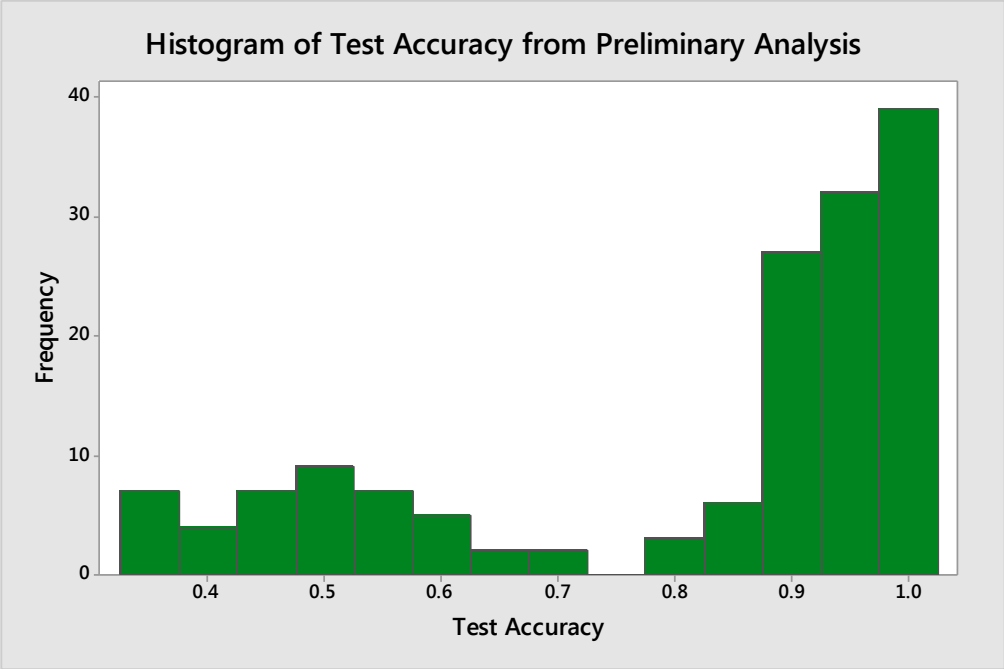

Figure 17. Histogram showing the frequency of accuracy results for the test set: preliminary analysis

The results obtained from this preliminary set of tests are promising. They led to the selection of a refined set of model inputs to run a large number of iterations for further testing. There were five combinations of batch size and number of epochs that outperformed others in this preliminary study. These input values were selected and each combination was run through 100 iterations (500 total runs). The model input values used for further analysis are shown below.

$$(\text{Batch Size, Number of Epochs}) \in \{(8,15), (8,20), 8,25), (16,5), (20,5)\}$$

These executions were performed much the same way as those in the preliminary analysis. The results of these experiments are discussed in the following section.

5. Results

Upon further experimentation with the five combinations of model inputs specified above, the model again produced promising results. A similar clustering pattern is visible in the scatter plot shown in Figure 18 to the one shown above in Figure 14. Again, we see strong evidence for a positive correlation between the validation and test accuracy scores. This further supports the assertion made earlier that a strong validation accuracy should produce a similarly strong test accuracy. With the exception of a few points being located on the test accuracy boundary between the two clusters, there is no overlap between the groups. In this plot, a validation accuracy score less than 70% would indicate a high probability for receiving a test score below the same mark. Conversely, a validation accuracy greater than 70% would have a high likelihood to result in a test accuracy greater than 80%. Tables 11-14 in Appendix 3 provide a full summary of the average accuracy scores and their standard deviations from the 100 iterations across all five input combinations.

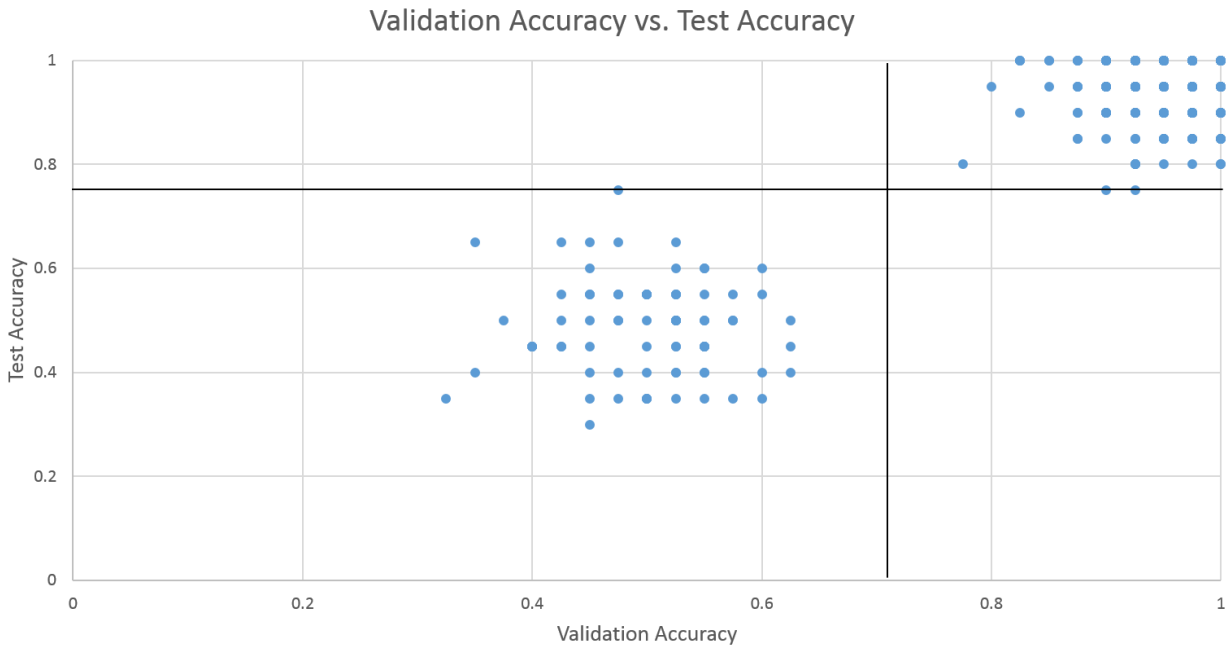

Figure 18. Scatter plot displaying the results of comparing the validation accuracy vs. the test accuracy from the large experiment

The statistics presented by these data points are also compelling. The overall experiment accuracy statistics generated by the 500 iterations are lower than those corresponding to the same settings presented in the preliminary analysis. However, this is not surprising. There is a more likely chance to have a poor partition of images in 100 iterations of any combination of inputs than there is for 5 iterations of the same combinations. While the accuracy scores, on average, are slightly lower, the results still reach 88% for both the validation and test accuracies over the 500 iterations. Although the high-performance cluster looks smaller in size, the number of data points in this cluster far outnumbers the data points in the low-performance cluster. The basic statistics for the 500 iterations and for each cluster are shown below in Tables 4-6. A full summary of the results from this experiment may be seen in Appendix 3.

Table 4. Summary statistics for the entire pool of analysis runs

Statistic	Validation Accuracy	Test Accuracy
Count	500	500
Max	1	1
Min	0.325	0.3
Avg	0.881	0.878

Table 5. Summary statistics for the low-performance cluster results

Statistic	Validation Accuracy	Test Accuracy
Count	78	78
Max	0.625	0.75
Min	0.325	0.300
Avg	0.500	0.485

Table 6. Summary statistics reflecting the high-performance cluster results

Statistic	Validation Accuracy	Test Accuracy
Count	422	422
Max	1	1
Min	0.775	0.750
Avg	0.951	0.951

While the individual averages for each of the five combinations of model inputs are slightly lower than the results seen in the preliminary experiments, the average reflected for both accuracy measures above in Table 6 is very comparable to the levels shown in the preliminary work. Even more encouraging is the increase in the proportion of results contained in the high-performance cluster relative to the experiment size. The preliminary experiments showed 71.3% of results in the high-performance cluster, while the large experiments showed 84.4% in this cluster. For comparison, a surface plot and histograms of the results of the large experiments are shown below in Figures 19-21. Another encouraging result of the large experiment is the reduction of noise in the foreground of the surface plot. The smaller proportion of results appearing in the low-performance cluster means a stronger pool of model inputs were tested in the large experiment. Ideally, the only points rising above zero in this surface plot would be in the back corner—members of the high-performance cluster. However, the model is still subject to poor image partitions, resulting in both low validation and test accuracy scores.

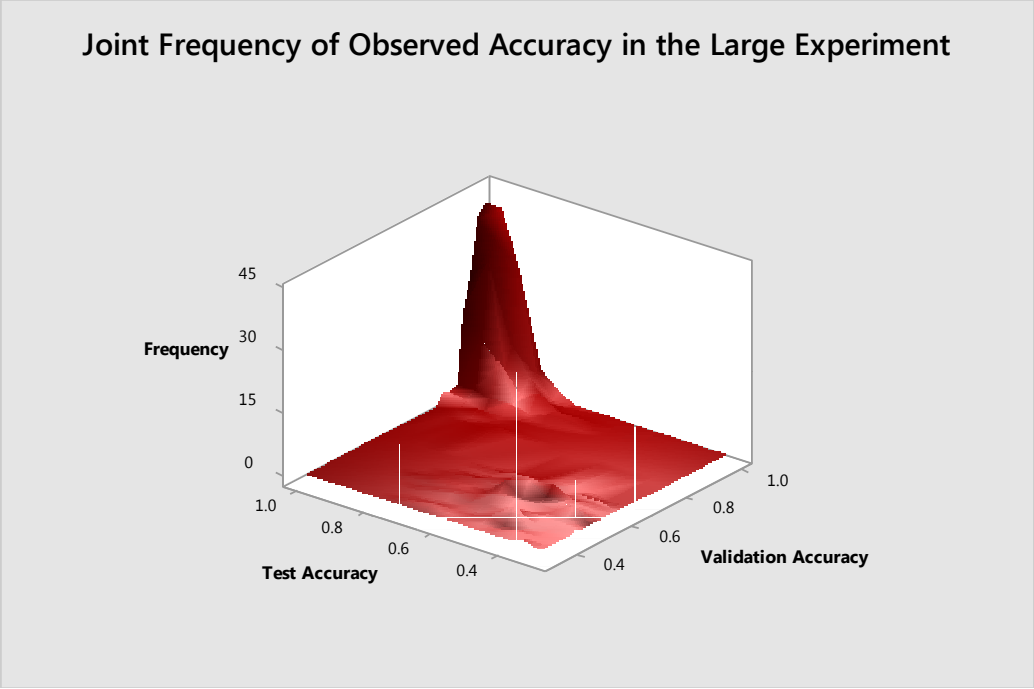

Figure 19. Surface plot showing the frequency of the different pairs of accuracy results for the validation and test sets for the large experiment

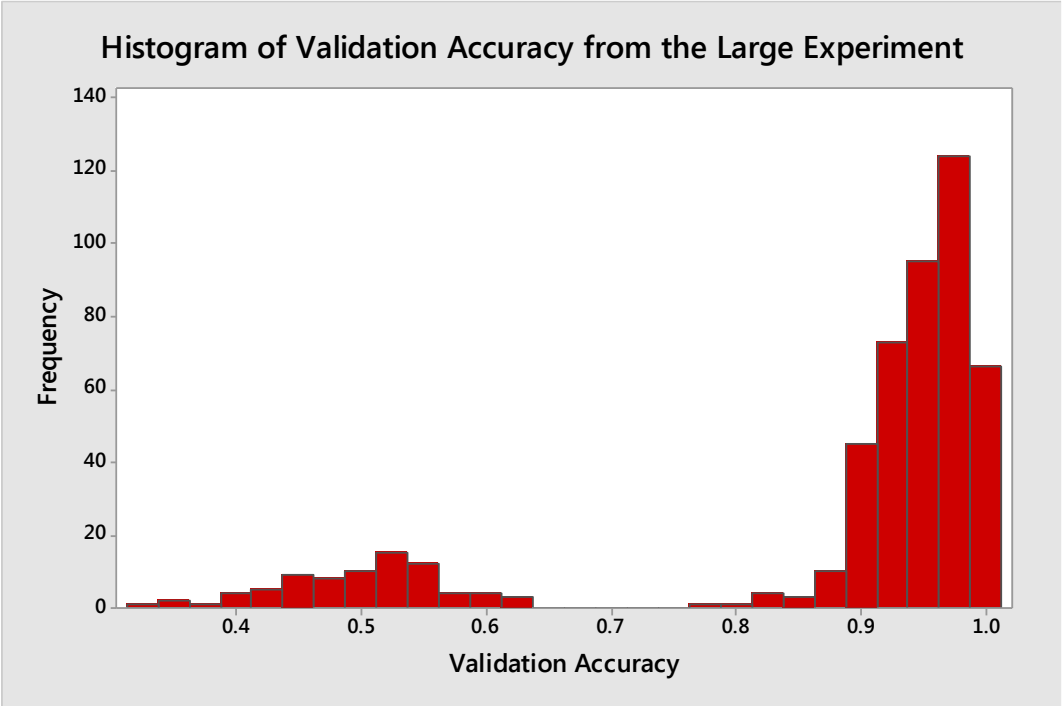

Figure 20. Histogram showing the frequency of accuracy results for the validation set for the large experiment

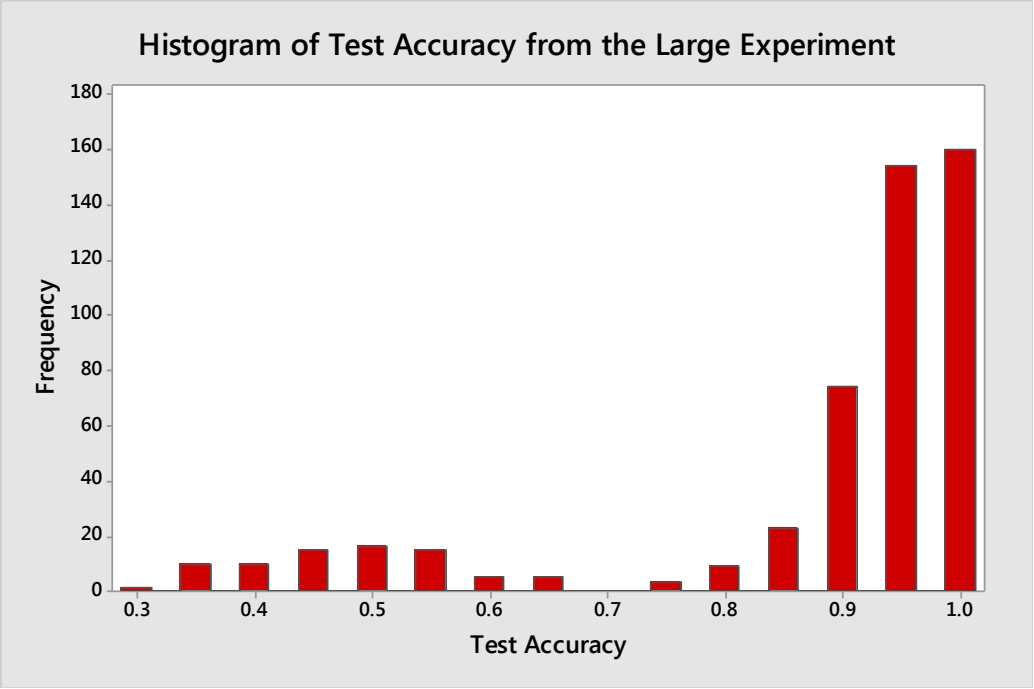

Figure 21 Histogram showing the frequency of accuracy results for the test set for the large experiment

In addition to the promising results presented above, it is important to understand how the model may be potentially improved. Because the VGG16 model is a proven, pre-trained model and we have done experiments to determine the best combinations of inputs for the dataset used in this research study, it is important to examine particular images in the dataset that seem to give the model difficulty. An approach used in this study to identify such areas for improvement was to examine the images included in the test set for points contained in the low-performance cluster. Specifically, the data points contained in the low-performance cluster for the best-performing set of model inputs from the large experiment (Batch size = 8, Number of Epochs = 15). Thirteen data points are contained in the low-performance cluster from this particular set of model inputs (i.e. 13 of the 100 iterations for this combination of model inputs resulted in poor validation and test accuracy scores).

Interestingly, the thirteen output files contained in the low-performance cluster for this particular combination of model inputs all possessed 'black or white' predictions for the test sets. By 'black and white' we mean the test set for that particular partition was either predicted to be entirely contour-levee fields or entirely precision-leveled fields. The poor performance observed for these thirteen particular output files can then be attributed to a poor random partition of images into the training and validation sets—the model parameter weights did not do a fine job discerning between the two classes of images in these instances. In each of the thirteen cases, the actual images contained in the test sets were distributed roughly half contour-levee images and half precision-leveled images, giving support to the hypothesis the parameter weights were not accurately assigned due to a poor partition of images in the training and validation sets. It is compelling the low-performance cluster always yields a model that predicts all images in the test set to be in the same classification. Again, this goes to support a poor random partition of images, and therefore, a poor assignment of model parameter weights.

With respect to the actual images contained in the test sets of these thirteen low-performance cluster data points, it is important to identify if any set of particular image files are giving the model difficulty in classification. Over the thirteen files, the names of the image files were recorded each time the model incorrectly predicted an image's class. 27 images were classified incorrectly more than once.

Five images were misclassified more than twice. A sample of the misclassified images is provided below in Figures 22-29 for reference. The figures below show images that were incorrectly classified at least twice over these thirteen iterations.

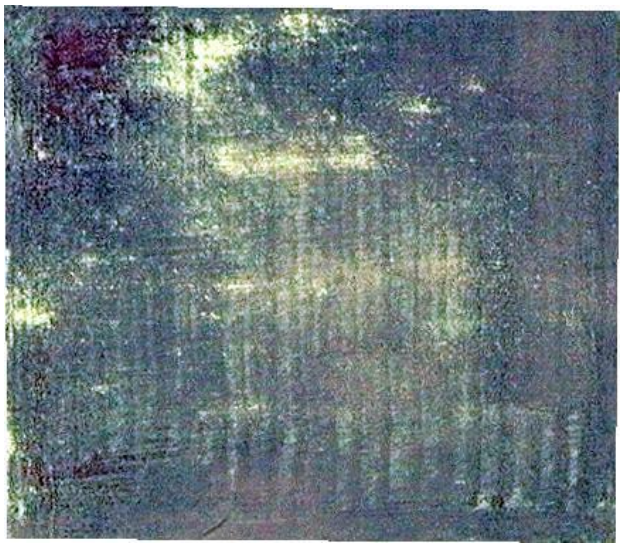

Figure 22. Precision-leveled field misclassified 4 times

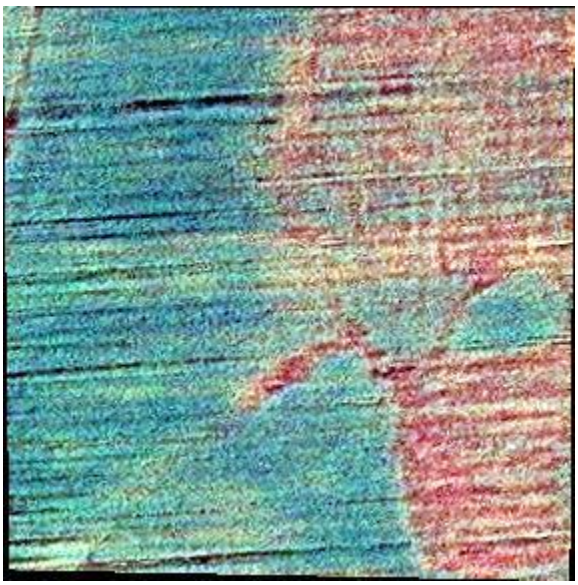

Figure 23. Precision-leveled field misclassified 3 times

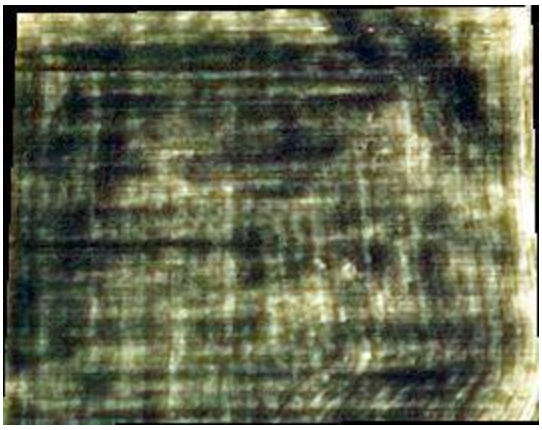

Figure 24. Precision-leveled field misclassified 3 times

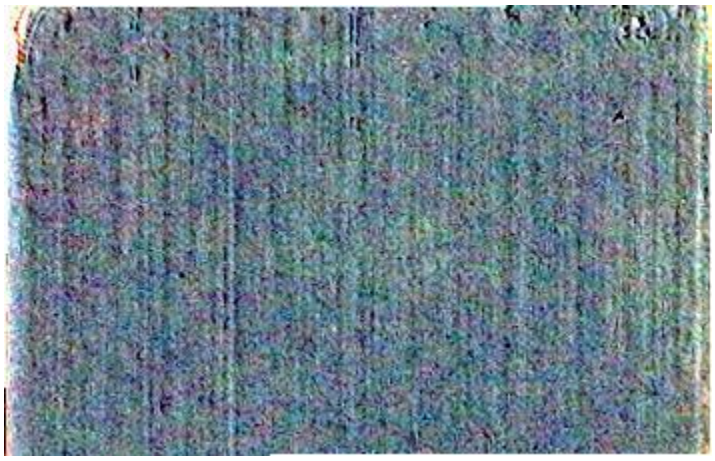

Figure 25. Precision-leveled field misclassified 3 times

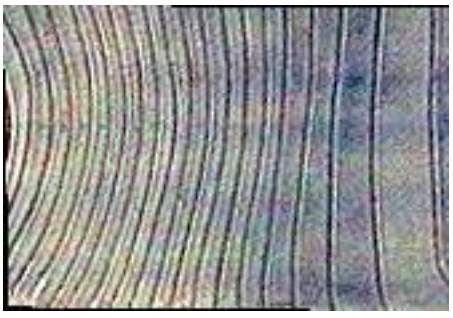

Figure 26. Contour-levee field misclassified 3 times

Figure 27. Contour-levee field misclassified 2 times

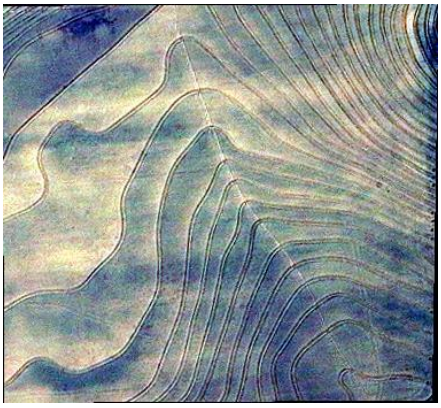

Figure 28. Contour-levee field misclassified 2 times

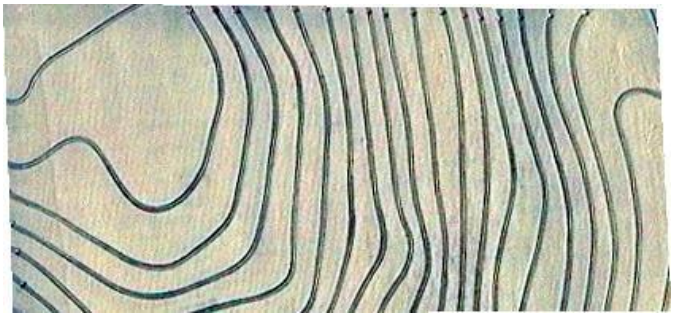

Figure 29. Contour-levee field misclassified 2 times

The eight images shown above in Figures 22-29 contain four precision-leveled fields and four contour-levee fields that were misclassified more than twice. The reason for this misclassification may be traced back to image quality. There are a few images which contain serious color distortions (a result of the ArcMap file export) and some which are not exactly rectangular due to the field's shape. There is likely a multitude of reasons behind the misclassification of these particular images, but the fact they appeared more than once in the misclassification category indicates some number of issues with the image. It is clear image quality is a critical component to the model's ability to correctly and confidently classify field images.

6. Conclusions and Future Research

The VGG16 neural network model is a proven model used for image classification. It is apparent through the results of this study it can be applied to field image datasets for the purposes of image classification. The results of this study indicate the dynamic database of field images classified by leveling type can certainly be created using neural network image classification techniques.

In the preliminary analysis conducted in this study, five combinations of model inputs were identified as promising candidates for further investigation:

$$(\text{Batch Size, Number of Epochs}) = \{(8,15), (8,20), (8,25), (16, 5), (20,5)\}.$$

These combinations were then subjected to further experimentation (100 repetitions each), resulting in an overall model accuracy classification of 88% for both validation and test accuracy statistics. The model input combination Batch size = 8, Number of Epochs = 15 produced the top accuracy score of 89.4% and 89.7% for the validation and test accuracy statistics, respectively. While this accuracy level is certainly not state-of-the-art, it demonstrates neural networks hold the potential to accurately classify field imagery.

The results of this study are promising, but there is much room for growth and future work on this topic. Principally, the increased collection of high-quality field images to expand the dataset used to fine-tune this model. As the dataset continues to be populated with quality images of rice fields, the model will have more images to discern differences in the field patterns, and ultimately move closer and closer to state-of-the-art accuracy levels for image classification. In the future, it will be important to continue the data collection effort on this topic to enable and incentivize the successful implementation of sustainable growing methods.

Furthermore, it would seem the main reason for the low-performance cluster data points observed in this study is a poor random partition of images from the main dataset into the training, validation, and testing sets. In order to combat this issue, a modification to the random partition code could be made. If this code were to be modified so an equal number of randomly selected images from each classification populated each of the three sets, this random partition error might be reduced (e.g. instead of sampling 140 random images for the training data set from one pool, randomly select 70 contour-levee image files and 70 precision-leveled image files to populate this set). It would at least be worthwhile to investigate the effects this type of random partitioning has on the model's ability to accurately.

It also would be valuable to investigate other neural network models' abilities to classify rice field imagery. While VGG16 is a proven image classification model, it has been surpassed in recent years on accuracy measures. It would be worth looking into other, more-recent neural network architectures to see if any additional advantages could be realized in classification accuracy or computational efficiency. Additionally, it would be interesting to examine a neural network's ability to classify imagery of other agricultural field types outside of the rice sector. The applications of effectively classifying other field crops could provide farmers with further financial or environmental advantages.

Finally, as neural network and deep learning technology continues to advance, it would be interesting to examine the effective differences between different python libraries and their abilities to effectively classify field imagery. An up and coming example (mentioned previously) is TensorFlow. TensorFlow is relatively young in its development stages, but is predicted to be extremely powerful once

it becomes more widely-known and more widely-implemented. Theano is currently the more prevalent of the two deep learning libraries mentioned in this study, but there is certainly the possibility for an improvement in classification accuracy or in computational efficiency when executing these neural network image classification models.

Works Cited

[1] J. Schewe, D. G. J. Heinkea, I. Haddeland, N. Arneilid and D. C. e. al., "Multimodel assessment of water scarcity under climate change," *Proceedings of the National Academy of Science of the United States of America,* vol. 111, no. 9, pp. 3245-3250, 4 March 2014.

[2] L. Nalley, B. Linquist, K. Kovacs and M. Anders, "The Economic Viability of Alternative Wetting and Drying Irrigation in Arkansas Rice Production," *Agronomy Journal,* vol. 107, no. 2, pp. 579-587, 2015.

[3] N. Childs and J. Livezey, "Rice Backgrounder," United States Department of Agriculture, 2006.

[4] D. K. Ray, N. D. Mueller, P. C. West and J. A. Foley, "Yield Trends Are Insufficient to Double Global Crop Production by 2050," *PLOS One,* vol. 8, no. 6, 2013.

[5] R. Rafique, M. Peichl, D. Hennessy and G. Kiely, "Evaluating Management Effects onNitrous Oxide Emissions from Grasslands Using the Process-Based DeNitrification-DeComposition (DNDC) Model," *Elsevier,* pp. 1-11, 26 July 2011.

[6] M. W. Rosegrant, M. Ewing, G. Yohe, I. Burton, S. Huq and R. Valmonte-Santos, "Climate Change and Agriculture Threats and Opportunities," *Climate Protection Programme for Developing Countries,* 2008.

[7] Arkansas Natural Resources Commission, "Arkansas Groundwater Protection and Management Report for 2012," Arkansas Natural Resources Commission, Little Rock, 2012.

[8] C. M. Pittelkow, M. A. Adviento-Borbe, C. V. Kessel, J. E. Hill and B. A. Linquist, "Optimizing rice yields while minimizing yield-scaled global warming potential," *Global Change Biology,* pp. 1-12, 2014.

[9] L. Guerra, S. Bhuiyan, T. Tuong and a. R. Barker, Producing More Rice from Less Water from Irrigated Systems, Colombo: Water Management Institution, 1998.

[10] B. A. M. Bouman, R. M. Lampayan and T. P. Toung, Water Management in Irrigated Rice: Coping with Water Scarcity, Metro Manila: International Rice Research Institute, 2007.

[11] Intergovernmental Panel on Climate Change , "IPCC Guidelines for National Greenhouse Gas Inventories," Intergovernmental Panel on Climate Change , Hayama, 2006.

[12] University of Arkansas Rice Research and Extension Center, "Rice Research and Extension Center," 2016. [Online]. Available: https://aaes.uark.edu/research-locations/rice.aspx. [Accessed 18 September 2016].

[13] J. A. Hignight, K. B. Watkins and M. M. Anders, "Economic Analysis of Zero-Grade Rice and Land Tenure," *2009 Journal of the ASFMRA,* pp. 143-152, 2009.

[14] J. Epting, "Water Use in the Mississippi Delta - 2004 Report," YMD Joint Water Management District Annual Report, 2004.

[15] B. A. Linquist, M. M. Anders, M. A. A. Adviento-Borbe, R. L. Chaney, L. L. Nalley, E. F. F. D. Rosa and C. V. Kessel, "Reducing Greenhouse Gas Emissions, Water Use, and Grain Arsenic Levels in Rice Systems," *Global Change Biology,* pp. 1-11, 2014.

[16] A. L. Samuel, "Some Studies in Machine Learning Using the Game of Checkers," *IBM Journal of Research and Development,* 1959.

[17] D. T. Pham, "Machine learning techniques and their applications in manufacturing," *Proceedings of the Institution of Mechanical Engineers,* vol. 219, pp. 395-412, 2004.

[18] A. Dubey, "Applications of machine learning: cutting edge technology in HIV diagnosis, treatment and further research," *Computational Molecular Biology,* vol. 6, no. 3, pp. 1-6, 2016.

[19] P. Kamavisdar, "A survey on image classification approaches and techniques," *International journal of advanced research in computer and communication engineering,* vol. 2, no. 1, pp. 1005-1009, 2013.

[20] D. Lu, "A survey of image classification methods and techniques for improving classification performance," *international journal of remote sensing,* vol. 28, no. 5, pp. 823-870, 2007.

[21] S. B. Kotsiantis, "Supervised machine learning: a review of classification techniques," *Informatica,* pp. 249-268, 2007.

[22] M. Anthony, Neural Network Learning: Theoretical Foundations, Cambridge University Press, 2009.

[23] J. Howard, "fastai/courses/deeplearning1," FastAI, 2016. [Online]. Available: https://github.com/fastai/courses/tree/master/deeplearning1/nbs. [Accessed January-March 2017].

[24] Koutroumbas, Konstantinos, Theodoridis and Sergios, "The Perceptron Algorithm," in *Pattern Recognition*, Elsevier Science, 2008, pp. 93-103.

[25] K. Simonyan, "Very Deep Convolutional Networkf for Large-Scale Image Recognition," in *ICLR*, 2015.

[26] A. Krizhevsky, "ImageNet Classification with Deep Convolutional Neural Networks," *Neural Information Processing Systems,* pp. 1-9, 2012.

[27] A. Karpathy, "Convolutional Neural Networks for Visual Recognition," Stanford University, August 2016. [Online]. Available: http://cs231n.github.io/convolutional-networks/. [Accessed 7 March 2017].

[28] Q. Li, G. Huo, Y. Zhou and L. Zhou, "Image classification Using Biomimetic Pattern Recognition with Convolutional Neural Networks Features," *Computational Intelligence and Neuroscience,* vol. 2017, pp. 1-12, 2017.

[29] F. Chollet, "How convolutional neural networks see the world," The Keras Blog, 30 January 2016. [Online]. Available: https://blog.keras.io/how-convolutional-neural-networks-see-the-world.html. [Accessed 7 March 2017].

[30] National Agricultural Imagery Program, "NAIP Imagery," United States Department of Agriculture, 2016. [Online]. Available: https://www.fsa.usda.gov/programs-and-services/aerial-photography/imagery-programs/naip-imagery/index. [Accessed February 2017].

[31] United States Department of Agriculture, "Innovative Project Curbs Greenhouse Gas Emissions From Mid-South Rice Fields," 19 May 2016. [Online]. Available: http://www.nrcs.usda.gov/wps/portal/nrcs/blogdetail/nrcsblog/home/?cid=NRCSEPRD1047408. [Accessed 28 August 2016].

Appendix 1—Steps to Obtain Images from ArcMap

1. Navigate to the ‘Customize’ tab, click the ‘Extensions’ option and ensure the box around ‘Spatial Analyst’ is checked. This license’s permissions are needed to execute certain features of the program.
2. Again navigate to the ‘Customize’ tab, and hover over the ‘Toolbars’ option. Scroll down to enable the ‘Editor’ toolbar.
3. Create a shapefile in the appropriate directory. The shapefile should be given a unique name and should be classified as a ‘Polygon’ in the drop down menu.
4. Locate the ‘Editor’ toolbar again. Click the ‘Editor’ drop down option and choose ‘Start Editing’.
5. The navigation pane on the right side of the screen will pull up a list of all the active shapefiles contained in the geodatabase. Select the shapefile you wish to edit.
6. Draw the polygonal shape on top of the background imagery. In this case, draw rectangles on top of the different rice fields. Whenever the rectangle is complete, again click the ‘Editor’ drop down from the ‘Editor’ toolbar. Choose ‘Save Edits’ and then ‘Stop Editing’.
7. Activate the ‘Search’ tab on the navigation pane to the right. Search for ‘Extract by Mask’. In order for this feature to properly execute, the spatial analyst license permissions must be enabled (see step 1).
8. Click this tool. Three input fields will appear. Click the drop down arrow for the first field. Choose the name of the background imagery file. Click the drop down arrow for the second field. Click the name of the shapefile you wish to extract from the background imagery. Click the button to the right of the third input field. Navigate to the appropriate directory for storing the extracted image file. Name the file appropriately, ensuring to include the extension ‘.tif’ at the end of the file name. If the ‘.tif’ is not included, the Extract by Mask feature will not execute properly.
9. Once the image has been extracted, the file should appear in the navigation pane to the left (‘Name.tif’). Right click this file name. Hover over ‘Data’ option and then choose ‘Export Data’. A dialogue box will appear. Check the ‘Use Renderer’ box. A dropdown list in the bottom right will allow you to choose which file type you would like to generate. Choose ‘JPG’. Modify the name of the file as needed. Note: ensure the extension is capitalized (e.g., ‘Name.JPG’). The capitalization is important for the CNN’s interpretation of the image. Click ‘Save’.

Appendix 2—Pivot Table Results from Preliminary Analysis

Table 7. Test Accuracy results from preliminary analysis. Each cell is an average over the five repetitions performed for each combination of inputs. The Highlighted cells indicate the most promising input combinations.

Average of Test Accuracy	No. Epochs					
Batch Size	5	10	15	20	25	Grand Total
2	0.800	0.770	0.690	0.600	0.710	**0.714**
4	0.830	0.690	0.840	0.850	0.880	**0.818**
8	0.780	0.870	0.950	0.910	0.930	**0.888**
12	0.830	0.800	0.880	0.860	0.720	**0.818**
16	0.960	0.780	0.740	0.700	0.800	**0.796**
20	0.940	0.840	0.790	0.860	0.890	**0.864**
Grand Total	**0.857**	**0.792**	**0.815**	**0.797**	**0.822**	**0.816**

Table 8. Test Accuracy results from preliminary results. Each cell is the standard deviation of the five repetitions' accuracy results performed for each combination of inputs. The highlighted cells indicate the most promising input combinations from the average accuracy statistics shown in the above table.

StdDev of Test Accuracy	No. Epochs					
Batch Size	5	10	15	20	25	Grand Total
2	0.215	0.251	0.207	0.203	0.258	**0.220**
4	0.220	0.256	0.275	0.226	0.160	**0.221**
8	0.280	0.211	0.071	0.042	0.067	**0.162**
12	0.186	0.267	0.214	0.204	0.297	**0.224**
16	0.042	0.266	0.295	0.255	0.209	**0.230**
20	0.042	0.277	0.275	0.258	0.167	**0.210**
Grand Total	**0.184**	**0.240**	**0.231**	**0.222**	**0.207**	**0.216**

Table 9. Validation Accuracy results from preliminary analysis. Each cell is an average over the five repetitions performed for each combination of inputs. The Highlighted cells indicate the most promising input combinations.

Average of Validation Accuracy	No. Epochs					
Batch Size	5	10	15	20	25	Grand Total
2	0.745	0.745	0.665	0.590	0.765	**0.702**
4	0.870	0.670	0.910	0.865	0.875	**0.838**
8	0.770	0.860	0.960	0.955	0.955	**0.900**
12	0.885	0.865	0.870	0.880	0.760	**0.852**
16	0.960	0.800	0.750	0.670	0.785	**0.793**
20	0.945	0.855	0.775	0.865	0.850	**0.858**
Grand Total	**0.863**	**0.799**	**0.822**	**0.804**	**0.832**	**0.824**

Table 10. Validation Accuracy results from preliminary results. Each cell is the standard deviation of the five repetitions' accuracy results performed for each combination of inputs. The highlighted cells indicate the most promising input combinations from the average accuracy statistics shown in the above table.

StdDev of Validation Accuracy	No. Epochs					
Batch Size	5	10	15	20	25	Grand Total
2	0.256	0.293	0.241	0.207	0.276	**0.244**
4	0.196	0.285	0.174	0.177	0.199	**0.211**
8	0.237	0.230	0.022	0.037	0.048	**0.157**
12	0.192	0.151	0.193	0.215	0.284	**0.199**
16	0.034	0.229	0.320	0.281	0.296	**0.251**
20	0.060	0.200	0.226	0.204	0.266	**0.194**
Grand Total	**0.186**	**0.227**	**0.222**	**0.225**	**0.233**	**0.217**

Appendix 3—Pivot Table Results from Large Experiment

Table 11. Test Accuracy results from the large experiment. Each cell is an average over the 100 repetitions performed for each combination of inputs. The Highlighted cells indicate the most promising input combinations.

Average of Test Accuracy	Number of Epochs				
Batch Size	5	15	20	25	Grand Total
8		0.897	0.873	0.871	0.880
16	0.865				0.865
20	0.884				0.884
Grand Total	**0.874**	**0.897**	**0.873**	**0.871**	**0.878**

Table 12. Test Accuracy results from preliminary results. Each cell is the standard deviation of the 100 repetitions' accuracy results performed for each combination of inputs. The highlighted cells indicate the most promising input combinations from the average accuracy statistics shown in the above table.

StdDev of Test Accuracy	Number of Epochs				
Batch Size	5	15	20	25	Grand Total
8		0.165	0.187	0.190	0.181
16	0.188				0.188
20	0.167				0.167
Grand Total	**0.178**	**0.165**	**0.187**	**0.190**	**0.179**

Table 13. Validation Accuracy results from the large experiment. Each cell is an average over the 100 repetitions performed for each combination of inputs. The Highlighted cells indicate the most promising input combinations.

Average of Validation Accuracy	Number of Epochs				
Batch Size	5	15	20	25	Grand Total
8		0.894	0.877	0.881	0.884
16	0.867				0.867
20	0.885				0.885
Grand Total	**0.876**	**0.894**	**0.877**	**0.881**	**0.881**

Table 14. Validation Accuracy results from preliminary results. Each cell is the standard deviation of the 100 repetitions' accuracy results performed for each combination of inputs. The highlighted cells indicate the most promising input combinations from the average accuracy statistics shown in the above table.

StdDev of Validation Accuracy	Number of Epochs				
Batch Size	5	15	20	25	Grand Total
8		0.148	0.172	0.182	0.167
16	0.182				0.182
20	0.163				0.163
Grand Total	**0.173**	**0.148**	**0.172**	**0.182**	**0.169**